ViewSplat: View-Adaptive 3D Gaussian Splatting for Feed-Forward Synthesis

Moonyeon Jeong[1], Seunggi Min[1], Suhyeon Lee[2,†], and Hongje Seong[1,†]

[1] University of Seoul [2] Korea Electronics Technology Institute
{jmy06051,minsk0725,hjseong}@uos.ac.kr suhyeon@keti.re.kr
https://cvlab-uos.github.io/ViewSplat

Abstract. We present ViewSplat, a view-adaptive 3D Gaussian splatting network for novel view synthesis from unposed images. While recent feed-forward 3D Gaussian splatting has significantly accelerated 3D scene reconstruction by bypassing per-scene optimization, a fundamental fidelity gap remains. We attribute this gap to the limited capacity of single-step feed-forward networks to regress static Gaussian primitives that satisfy all viewpoints. To address this limitation, we shift the paradigm from static primitive regression to view-adaptive splatting. Instead of a rigid Gaussian representation, our pipeline learns a view-adaptive latent representation. Specifically, ViewSplat initially predicts base Gaussian primitives alongside the weights of scene-conditioned View MLPs. During rendering, these MLPs take target-view coordinates as input and predict view-dependent residual updates for each Gaussian attribute (*i.e.*, 3D position, scale, rotation, opacity, and color). This mechanism, which we term view-adaptive splatting, allows each primitive to rectify initial estimation errors, effectively capturing high-fidelity appearances. Extensive experiments demonstrate that ViewSplat achieves state-of-the-art fidelity while maintaining fast inference and real-time rendering; our large backbone variant runs at 15 FPS during inference and 90 FPS during rendering.

Keywords: Novel View Synthesis · 3D Gaussian Splatting · Pose-Free Reconstruction

1 Introduction

3D Gaussian Splatting (3DGS) [16] has recently emerged as a powerful representation in 3D computer vision, enabling high-fidelity real-time rendering. While traditional 3DGS frameworks rely on per-scene optimization, recent efforts have focused on generalizable feed-forward architectures [3, 5, 29, 34, 38]. These models predict scene representations directly from a few input images, enabling instant 3D reconstruction of unseen scenes and high-quality novel view synthesis (NVS) without costly target-scene optimization.

[†]Co-corresponding authors.

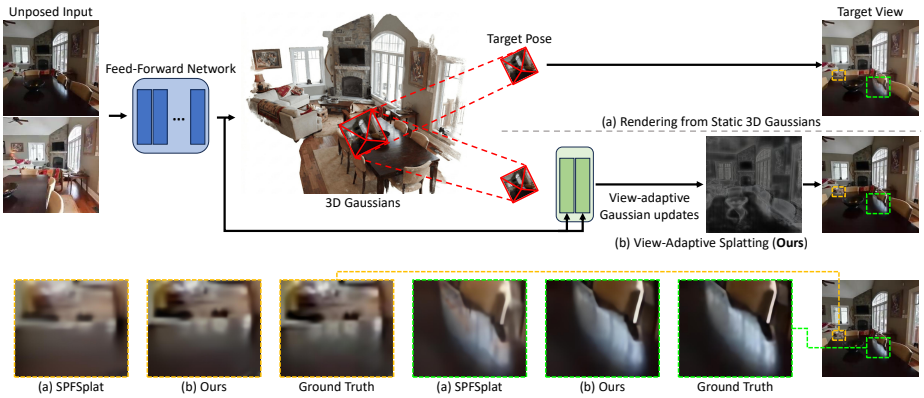


Fig. 1: Given unposed images as input, the feed-forward network reconstructs 3D Gaussians. While (a) rendering from static 3D Gaussians often suffers from blurred details, (b) our method introduces view-adaptive Gaussian updates conditioned on the target-view pose to refine the Gaussians on the fly. As shown in the bottom panels, our approach improves the reconstruction of fine-grained details (*e.g.*, sharp edges and specular reflections) compared to SPFSplat [10].

A major bottleneck in these generalizable pipelines is the heavy reliance on accurate camera poses, typically precomputed via Structure-from-Motion (SfM) [26]. To address this, recent works [10, 11, 15, 27] leverage 3D foundation models—such as DUSt3R [32], MASt3R [17], and VGGT [31]—to facilitate pose-free reconstruction. These frameworks enable the joint estimation of camera parameters and 3D Gaussian primitives [10, 13], delivering robust 3D reconstructions directly from uncalibrated, in-the-wild images.

Despite these advancements, a significant fidelity gap remains between feed-forward approaches and optimization-based methods [14, 21, 23, 35, 42]. We attribute this gap to the limited capacity of feed-forward networks to regress static Gaussian primitives that satisfy all potential viewpoints. These models typically employ a single-step regression to predict all Gaussian attributes (*i.e.*, 3D position, scale, rotation, opacity, and spherical harmonics (SH) coefficients). However, regressing optimal representations is an ill-posed task for a single forward pass, especially when dealing with complex geometries or non-Lambertian surfaces. While optimization-based methods can refine these attributes over thousands of iterations to capture sharp specularities and view-dependent SH coefficients, feed-forward models lack the capacity to *rectify* their initial predictions during inference.

To address this limitation, we propose ViewSplat, a novel framework that shifts the paradigm from static primitive regression to view-adaptive splatting (Fig. 1). The key insight behind ViewSplat is that achieving high-fidelity 2D rendering from a *specific* view is relatively straightforward, provided the underlying 3D primitives can be tailored to that viewpoint. We leverage this observation by shifting the network's objective: instead of struggling to predict universal 3D Gaussians, the network focuses on generating view-specific updates that are far more feasible to

learn. This shift allows our model to rectify initial errors and capture complex view-dependent details that a static model would otherwise miss. Building upon a pose-free feed-forward architecture [10, 11], our model introduces a mechanism that updates Gaussian primitives for every target viewpoint on-the-fly.

Specifically, ViewSplat predicts two components for a given scene: a set of base Gaussian primitives and the weights of scene-conditioned View MLPs. Unlike standard MLPs with fixed parameters, these View MLPs are conditioned on the input scene context. During the rendering stage, these MLPs take the target-view coordinates as input and predict view-dependent residual updates for all Gaussian attributes, including 3D position, scale, rotation, opacity, and SH coefficients. This allows each primitive to undergo geometric and photometric refinement tailored to the specific viewpoint. Consequently, ViewSplat can effectively compensate for initial estimation errors and synthesize high-frequency, non-Lambertian effects that are typically lost in static feed-forward representations. Our approach maintains the efficiency required for practical applications, achieving fast inference (15 FPS) and high-speed rendering (90 FPS) for the large-backbone variant while significantly outperforming existing generalizable models. Our main contributions are summarized as follows:

- We identify the static primitive bottleneck in feed-forward 3DGS and propose view-adaptive representations for high-fidelity synthesis.
- We introduce ViewSplat, which utilizes scene-conditioned View MLPs to predict view-dependent residuals for all Gaussian attributes, enabling the model to rectify geometric and appearance errors on-the-fly.
- We demonstrate that ViewSplat achieves state-of-the-art performance on standard benchmarks, effectively capturing complex view-dependent effects without the need for costly per-scene optimization or precomputed poses.

2 Related Work

2.1 Feed-Forward 3D Gaussian Splatting

The landscape of 3DGS [16] has recently diversified, with significant research efforts increasingly directed toward generalizable feed-forward architectures that enable instant 3D reconstruction and NVS. Early generalizable 3DGS methods are pose-required, relying on accurate camera parameters to aggregate multi-view features or construct cost volumes. For instance, pixelSplat [3] and MVSplat [5] leverage epipolar transformers for depth estimation and Gaussian primitive prediction. Similarly, large-scale models such as LGM [29], GRM [34], and GS-LRM [38] encode camera poses via Plücker ray embeddings. While effective, such methods rely on precomputed poses through SfM [26], which is computationally expensive and often unreliable in sparse-view scenarios.

To overcome these constraints, pose-free feed-forward methods have emerged. Supervised pose-free models like Splatt3R [27] and NoPoSplat [37] predict 3D Gaussians in a canonical space without inference-time poses, yet they still require

ground-truth poses for training supervision. More recently, self-supervised approaches such as SelfSplat [15], PF3plat [8], SPFSplat [10], and SPFSplatV2 [11] have effectively eliminated this dependency. In particular, SPFSplat and SPFSplatV2 leverage large-scale 3D foundation model backbones, such as MASt3R [17] and VGGT [31] to jointly estimate camera poses and 3D geometry in a unified end-to-end pipeline.

Despite these advancements, existing feed-forward 3DGS models are fundamentally constrained by static primitive regression. By predicting a fixed set of Gaussian attributes in a single pass, these methods struggle to reconcile conflicting visual information across diverse viewpoints, leading to blurred details and a failure to capture non-Lambertian effects like sharp specularities. Our ViewSplat breaks this bottleneck by shifting from rigid regression to view-adaptive refinement. Built upon the self-supervised, pose-free feed-forward frameworks [10, 11], we introduce a view-dependent update mechanism that *rectifies* Gaussian parameters on the fly for each target viewpoint, enabling high-fidelity synthesis that was previously only attainable through per-scene optimization.

2.2 View-Dependent 3D Representation

Modeling view-dependent effects is a long-standing challenge in neural rendering. In Neural Radiance Fields (NeRF) [24], the standard approach involves conditioning the radiance MLP on the viewing direction to capture view-dependent color. However, this often struggles to represent complex surface properties, particularly non-Lambertian effects and sharp specularities. To address this limitation, several works [1, 2, 28, 40] decompose visual appearance into intrinsic properties—such as lighting and materials—typically parameterized via Bidirectional Reflectance Distribution Functions (BRDFs). Ref-NeRF [30] further enhances specular fidelity by utilizing reflection vectors. While effective, these implicit methods often require auxiliary geometric priors, such as those derived from signed distance functions (SDFs) [18, 22], to obtain high-quality normals for accurate shading.

The introduction of 3DGS [16] has opened new avenues for explicit view-dependent modeling. While 3DGS originally employs low-order SH to represent view-dependent color, SH coefficients frequently lack the frequency bandwidth necessary to capture sharp specularities. To overcome this limitation, recent extensions like GaussianShader [14] and RTR-GS [42] integrate explicit shading models into the Gaussian framework. Other approaches, such as Spec-Gaussian [35], introduce anisotropic spherical Gaussians to provide more flexible parameterization than traditional SH. MirrorGaussian [21] tackles the specific challenge of mirror reflections by incorporating symmetry-based Gaussian reflections. Beyond appearance, Scaffold-GS [23] explores structural stabilization of Gaussian attributes across viewpoints using an anchor-based representation.

While these advancements have significantly improved the fidelity of view-dependent rendering, they remain predominantly tied to per-scene optimization. This dependency results in high computational costs and a lack of generalization to unseen scenes. In contrast, ViewSplat addresses view dependence within a feed-forward framework by introducing viewpoint-conditioned attribute modulation.

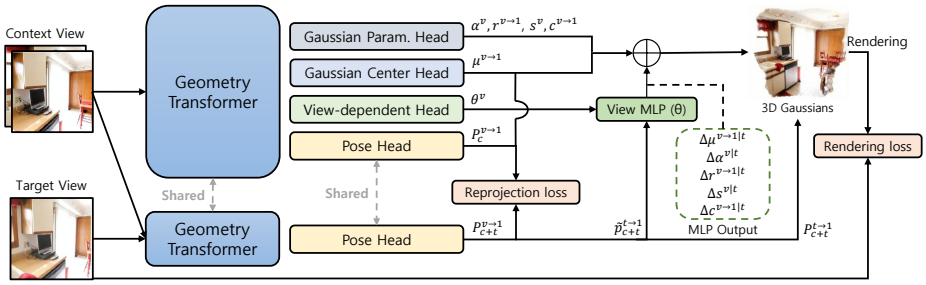


Fig. 2: Architecture of ViewSplat. Built upon a shared geometry transformer backbone [17, 31], our framework simultaneously predicts canonical 3D Gaussians and camera poses from unposed images using Gaussian heads and a pose head. To accurately capture view-dependent effects, a view-dependent head generates per-pixel View MLPs, which take the target-view pose as input to predict residual offsets for Gaussians. These offsets are then applied to refine the canonical Gaussians during rendering.

This allows for efficient, high-fidelity reconstruction and synthesis of complex view-dependent effects without the need for scene-specific training or costly iterative optimization.

3 Method

We propose ViewSplat, a view-dependent feed-forward 3D Gaussian splatting framework that jointly reconstructs 3D Gaussians and estimates camera poses directly from multi-view images without requiring ground-truth poses. Given a set of images, the network predicts the 3D Gaussian parameters in a canonical space defined by the first input image, which serves as the global reference coordinate frame. Simultaneously, it estimates the camera pose for each view, representing the canonical-to-camera transformation. Leveraging this relative pose information, our model predicts view-dependent residual offsets for the Gaussian parameters. These offsets are added to the canonical Gaussians to refine their attributes, enabling rendering of view-dependent effects such as specular highlights.

An overview of our architecture is illustrated in Fig. 2. We first present the preliminaries of our pose-free Gaussian reconstruction framework based on SPFSplat [10, 11] in Sec. 3.1. We then introduce our proposed view-dependent head in Sec. 3.2 and detail the view-dependent Gaussian refinement process in Sec. 3.3. Finally, we describe the training objective, including the rendering and reprojection losses, in Sec. 3.4.

3.1 Preliminaries

Our framework builds upon prior pose-free Gaussian reconstruction architectures, including SPFSplat and its subsequent variants [10, 11]. These models adopt geometry transformer backbones, such as MASt3R [17] and VGGT [31], along with pose and Gaussian prediction heads to reconstruct a 3D scene from N unposed images $\{I^v\}_{v=1}^N$.

Geometry Transformer. A ViT-based encoder-decoder architecture [6] is employed to extract multi-view feature tokens. The ViT encoder processes patchified RGB inputs through self-attention, after which the decoder enables cross-view interaction to aggregate geometric and appearance information across the N encoded views.

The specific interaction mechanism is determined by the backbone. MASt3R-based models employ pairwise cross-attention between views, whereas the VGGT-based V2-L variant performs global self-attention over a unified token sequence. In SPFSplatV2 and V2-L, a masked attention strategy is additionally introduced in the decoder to improve the efficiency of camera pose estimation and geometry reconstruction. The resulting tokens, enriched with global 3D context, serve as the foundation for both the baseline Gaussian reconstruction and our proposed view-dependent refinement.

Canonical 3D Gaussian Reconstruction. To reconstruct the 3D scene, two Dense Prediction Transformer (DPT)-based heads [25] predict 3D Gaussian parameters in a dense, per-pixel manner. The first DPT head, referred to as the Gaussian center head, processes the decoded tokens to predict per-pixel 3D coordinates that define the Gaussian centers. The second head, referred to as the Gaussian parameter head, estimates the remaining Gaussian attributes for each pixel. These 3D Gaussians are aligned in a canonical space, where the first input view I_1 serves as the global reference coordinate frame. Specifically, the predicted Gaussians are parameterized as:

$$\{\mathcal{G}^{v\to1}\}_{v=1,\dots,N} = \{(\mu_j^{v\to1}, \alpha_j^v, r_j^{v\to1}, s_j^v, c_j^{v\to1})\}_{j=1,\dots,H\times W}, \tag{1}$$

where each Gaussian primitive consists of a center position $\mu \in \mathbb{R}^3$, a rotation quaternion $r \in \mathbb{R}^4$, a scale vector $s \in \mathbb{R}^3$, an opacity value $\alpha \in \mathbb{R}$, and SH coefficients c. Following DPT, high-resolution skip connections are incorporated to preserve fine-grained spatial details from the input images.

Pose Estimation. For pose-free rendering, the pose head estimates the camera poses in a single feed-forward step. By taking the concatenated and pooled representations from the encoder and decoder, a lightweight MLP outputs a 10-dimensional relative pose comprising a 6D rotation representation and a 4D homogeneous translation vector $[t_x, t_y, t_z, 1]^\top$, following [4]. This pose is then converted into a homogeneous transformation matrix $P^{v\to1} \in \mathbb{R}^{4\times4}$, which is defined as:

$$P^{v\to1} = \begin{bmatrix} R^{v\to1} & t^{v\to1} \\ \mathbf{0}^\top & 1 \end{bmatrix}, \tag{2}$$

where $R^{v\to1} \in \mathbb{R}^{3\times3}$ is the rotation matrix and $t^{v\to1} \in \mathbb{R}^3$ is the translation vector. This matrix is the relative transformation of any input view I_v to the canonical reference view I_1. To establish a consistent global coordinate system, the first input view I_1 is defined as the origin, explicitly assigning its canonical

pose as:

$$P^{1\to 1} = \begin{bmatrix} I & \mathbf{0} \\ \mathbf{0}^\top & 1 \end{bmatrix}, \tag{3}$$

where I represents the identity matrix and $\mathbf{0}$ denotes the zero translation vector.

During training, to synthesize novel views without ground-truth poses, the model additionally predicts the poses of target views, denoted as $P^{t\to 1}$. As illustrated in Fig. 2, this target pose estimation leverages features from both context and target views to better capture the global scene structure. Importantly, to prevent information leakage, the initial Gaussian reconstruction is strictly conditioned on the context views, and remains independent of the target pose prediction.

3.2 View-Dependent Head

As described in Sec. 3.1, the baseline pipeline reconstructs a static set of Gaussians in the canonical space. However, due to its fixed geometry and the inherent limitations of SH, this static formulation often struggles to capture complex view-dependent effects, such as specular highlights, reflections, and subtle appearance variations. To address this limitation, we introduce a DPT-based view-dependent head that adopts a hypernetwork architecture [7] to predict per-pixel residual offsets for the Gaussian parameters. Instead of directly outputting offsets, this head takes feature tokens from the geometry transformer to capture rich geometric and material context, and generates pixel-wise parameters for a secondary network, which we refer to as a View MLP.

This design allows the model to generate pixel-specific response functions tailored to the local properties of the scene. The generated View MLPs are conditioned on the target camera pose and compute the corresponding Gaussian parameter offsets. By predicting these View MLPs on the fly, the architecture can flexibly adapt its refinement strategy to the specific requirements of the target viewpoint. For instance, the head can generate a more complex mapping for specular surfaces to capture shifting reflections, while maintaining simpler functions for diffuse regions. This separation between context-driven weight generation and pose-driven offset calculation enables the model to capture intricate high-frequency appearance variations without being constrained by a static predefined parameter space.

3.3 View-Dependent Gaussian Refinement

We now describe how the generated View MLPs utilize the target pose and refine the canonical Gaussians. The View MLP takes a 4D per-pixel target pose representation as input to compute Gaussian parameter offsets. This 4D pose representation, comprising a 3D unit viewing direction and a 1D distance term, is carefully derived from the target camera extrinsics $P^{t\to 1}$.

Target Pose Reparameterization. The raw target extrinsic $P^{t\to1} \in \mathbb{R}^{4\times4}$ is a homogeneous world-to-camera transformation matrix composed of a 3×3 rotation matrix and a 3D translation vector, as defined in Eq. (2). Directly feeding this full transformation matrix into the network is undesirable as the complex mathematical constraints inherent in transformation matrices (*e.g.*, rotation orthogonality) are difficult for an MLP to optimize. To alleviate this, we reparameterize the target pose into a compact and geometrically meaningful representation consisting of a 3D unit viewing direction and a 1D distance. This design reduces the input dimensionality from 16 scalar elements to 4, while retaining essential view-dependent cues, facilitating network learning. We detail the formal mathematical derivation of the target pose representation in the supplementary material.

Gaussian Refinement. Given the 4D target pose representation, the View MLP predicts parameter offsets for the Gaussians at each pixel. Specifically, the network outputs the following offset values:

$$\{\Delta\mathcal{G}^{v\to1}\}_{v=1,\dots,N} = \{(\Delta\mu_j^{v\to1|t}, \Delta\alpha_j^{v|t}, \Delta r_j^{v\to1|t}, \Delta s_j^{v|t}, \Delta c_j^{v\to1|t})\}_{j=1,\dots,H\times W}. \tag{4}$$

These offsets encode how each Gaussian, originating from a specific context view and pixel, should adapt when observed from the target viewpoint t. The predicted offsets are directly applied to the canonical 3D Gaussians defined in Eq. (1) via element-wise residual addition (*e.g.*, $\hat{\mu} = \mu + \Delta\mu$), yielding the final refined parameters $\{\hat{\mathcal{G}}^{v\to1}\}_{v=1,\dots,N}$. The rasterizer then renders the novel view from the target pose $P^{t\to1}$:

$$\hat{I}^t = \mathcal{R}(P^{t\to1}, \{\hat{\mathcal{G}}^{v\to1}\}_{v=1,\dots,N}). \tag{5}$$

3.4 Loss Function

Following the training strategy of SPFSplat [10], our model is optimized end-to-end without relying on ground-truth camera poses. The overall training objective consists of an image rendering loss $\mathcal{L}_{render}$ and a geometric reprojection loss $\mathcal{L}_{reproj}$. Specifically, $\mathcal{L}_{render}$ is formulated as a combination of MSE photometric loss and LPIPS [39] perceptual loss (with a weight of $\lambda_{LPIPS} = 0.05$) to balance low-level photometric accuracy and high-level perceptual similarity. To stabilize training in the canonical space and enforce explicit geometric constraints, we employ $\mathcal{L}_{reproj}$, which penalizes the distance between 2D pixel coordinates and the projected 3D Gaussian centers using estimated relative poses. The total loss is defined as:

$$\mathcal{L}_{total} = \mathcal{L}_{render} + \lambda_{reproj}\mathcal{L}_{reproj}, \tag{6}$$

where λ_{reproj} is a weighting factor (set to 0.001). We provide full mathematical derivations and further implementation details in the supplementary material.

4 Experiments

We evaluate the performance of our method on novel view synthesis and cross-dataset generalization across multiple datasets. In addition, we conduct comprehensive ablation studies to analyze the contribution of each component.

4.1 Experimental Settings

Datasets. We train and evaluate our method on the RealEstate10K (RE10K) [41], which contains large-scale real estate videos, and ACID [20], which consists of nature scenes captured by aerial drones. The camera poses for both datasets are obtained via SfM, but our method operates in a strictly pose-free manner, entirely bypassing the need for such precomputed information. We adopt the official train–test splits used in prior works [3, 5, 10, 37]. Furthermore, to validate cross-dataset generalization, we follow [5, 10, 37] and evaluate our model on ACID, the object-centric DTU dataset [12], and DL3DV [19]—a large-scale outdoor benchmark comprising 10K videos with complex camera trajectories that extend far beyond RE10K.

Baselines. We compare our method with several baselines for novel view synthesis. These baselines include pose-required models (pixelSplat [3] and MVSplat [5]), supervised pose-free models (CoPoNeRF [9], Splatt3R [27], and NoPoSplat [37]), and self-supervised models (SelfSplat [15], PF3plat [8], SPFSplat [10], and SPFSplatV2 [11]).

Evaluation Protocols. We assess the quality of novel view synthesis using standard metrics: pixel-level PSNR, patch-level SSIM [33], and feature-level LPIPS [39].

During NVS evaluation, target views are conventionally rendered using ground-truth poses [3, 5, 9, 27]. Alternative approaches include rendering with estimated target poses (*e.g.*, PF3plat [8], SelfSplat [15], SPFSplat [10], and SPFSplatV2 [11]) or employing an evaluation-time pose alignment (EPA) strategy (NoPoSplat [37]), which optimizes target-view poses during evaluation while freezing the reconstructed Gaussians. In our experiments, we evaluate our method using the estimated target poses. This setting allows us to jointly evaluate the reconstruction quality and the alignment between the estimated poses and the learned Gaussians.

4.2 Implementation Details

We implement ViewSplat in PyTorch, utilizing a CUDA-based 3DGS renderer with gradient support for camera poses. All models are trained on multiple NVIDIA RTX 4090 GPUs, while inference is performed on a single GPU.

Following prior works [10, 11], we adopt the geometry transformer backbones from MASt3R [17] for the SPFSplat and SPFSplatV2 architectures, and

Table 1: Performance comparison of novel view synthesis on RE10K [41] and ACID [20] datasets. We report the average metrics across all scenes. Results using the **ViewSplat** framework are highlighted in **bold**, demonstrating consistent performance gains across all SPFSplat variants. * indicates the use of the evaluation-time pose alignment (EPA) strategy.

Method	RE10K			ACID		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Pose-Required						
pixelSplat	23.859	0.808	0.184	25.889	0.780	0.194
MVSplat	24.012	0.812	0.175	25.561	0.775	0.195
Supervised Pose-Free						
CoPoNeRF	18.938	0.619	0.388	20.950	0.606	0.406
Splatt3R	18.688	0.337	0.596	18.060	0.510	0.407
NoPoSplat*	25.033	0.838	0.160	25.961	0.781	0.189
Self-Supervised Pose-Free						
SelfSplat	19.152	0.680	0.328	22.089	0.694	0.298
PF3plat	21.042	0.739	0.233	21.206	0.632	0.293
SPFSplat	25.484	0.847	0.153	26.070	0.781	0.186
+**ViewSplat**	**26.317**	**0.857**	**0.144**	**26.661**	**0.790**	**0.177**
SPFSplatV2	25.693	0.853	0.149	26.284	0.791	0.182
+**ViewSplat**	**26.468**	**0.861**	**0.140**	**26.873**	**0.801**	**0.173**
SPFSplatV2-L	25.668	0.855	0.137	26.674	0.806	0.162
+**ViewSplat**	**26.798**	**0.870**	**0.124**	**27.509**	**0.820**	**0.149**

VGGT [31] for the V2-L variant. To ensure stable convergence, we initialize the geometry transformer, Gaussian parameter head, Gaussian center head, and pose head using pretrained weights from the corresponding SPFSplat variants. The remaining component, our view-dependent head, is zero-initialized; this allows the network to initially preserve the baseline static reconstruction while progressively learning to incorporate view-adaptive refinements.

The view-dependent head generates pixel-wise View MLPs with a single hidden layer of 16 dimensions. All experiments are conducted at a resolution of 256×256, except for the V2-L variant, which follows its backbone's default resolution of 224×224.

Training Strategy. We adopt a frozen-backbone training strategy, where the pretrained weights of the base models (SPFSplat or SPFSplatV2) remain fixed while only the view-dependent head is optimized. We use the Adam optimizer with a learning rate of 1×10^{-4} and a batch size of 12. Each batch contains a single scene with its corresponding input and target views. Consistent with prior work, the frame distance between input views is progressively increased during training to facilitate curriculum learning.

4.3 Results

Novel View Synthesis. We evaluate ViewSplat on RE10K and ACID, and the quantitative results are summarized in Tab. 1. Our framework, particularly the V2-L variant, consistently achieves new state-of-the-art performance across

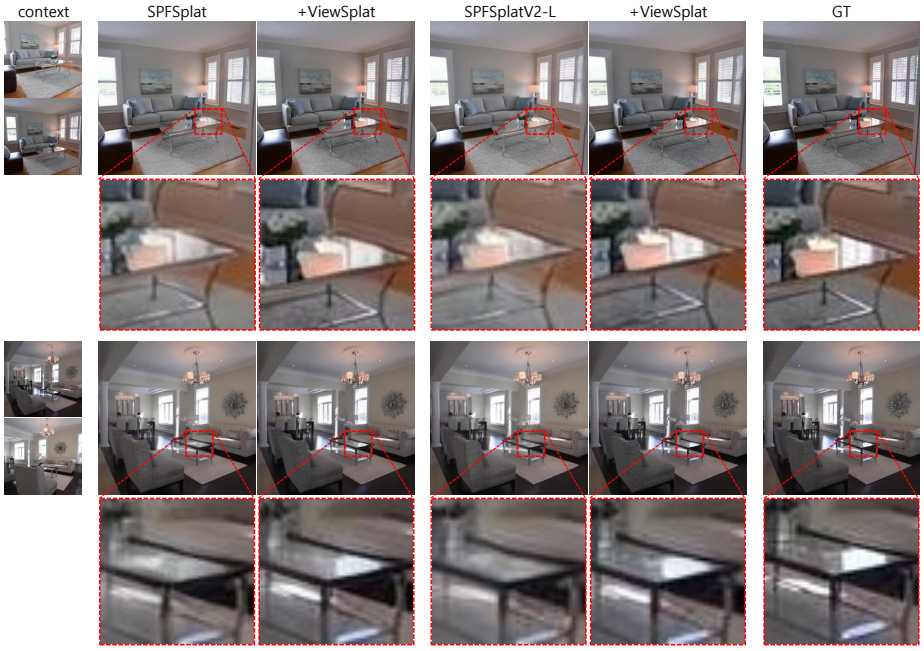


Fig. 3: Qualitative comparison on RE10K. Compared to baseline methods, our approach accurately reconstructs sharp reflections.

all metrics. A key finding is the orthogonal performance gain provided by our view-adaptive refinement. Across all backbone variants—SPFSplat, V2, and V2-L—ViewSplat consistently yields substantial improvements in rendering fidelity. This consistency underscores the scalability and robustness of our approach in capturing intricate view-dependent details across diverse environments, from structured interiors to complex natural landscapes. Qualitative comparisons in Fig. 3 further demonstrate ViewSplat's ability to synthesize high-frequency specularities and maintain geometric consistency, resulting in photorealistic novel views. We provide detailed results with varying levels of camera overlap and additional qualitative results in the supplementary material.

Table 2: Comparison of efficiency on a single NVIDIA RTX 4090 GPU using RE10K. ViewSplat is evaluated on three SPFSplat backbone variants.

Method	Params↓ (B)	Inference time↓ (s)	Rendering speed↑ (FPS)
SPFSplat	0.616	0.046	386
+ViewSplat	0.658	0.057	154
SPFSplatV2	0.613	0.039	390
+ViewSplat	0.655	0.052	149
SPFSplatV2-L	1.223	0.066	231
+ViewSplat	1.256	0.068	90

Table 3: Comparison of rendering performance between SPFSplat with varying SH degrees and the proposed ViewSplat on RE10K. "SPFSplat (Std.)" denotes the standard SPFSplat setting with SH degree 4. The **best** results are highlighted.

Method	SH Degree	PSNR↑	SSIM↑	LPIPS↓
SPFSplat (Std.)	4	25.486	0.847	0.153
SPFSplat	6	25.408	0.845	0.154
SPFSplat	8	25.418	0.845	0.153
+ViewSplat	4	**26.317**	**0.857**	**0.144**

Efficiency. We evaluate the computational efficiency of our framework on a single NVIDIA RTX 4090 GPU (Tab. 2). Integrating the ViewSplat module introduces minimal overhead to the forward pass, adding only 11–13 ms to jointly reconstruct the 3D Gaussians and predict the View MLP parameters.

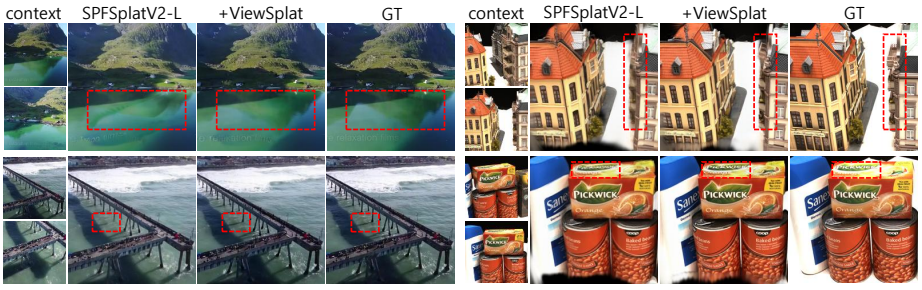
However, target-pose conditioning incurs a substantial drop in rendering speed (*e.g.*, from 386 to 154 FPS for the base backbone) since residual attributes must be re-evaluated for each novel viewpoint. Nevertheless, even with our largest SPFSplatV2-L backbone, ViewSplat comfortably maintains 90 FPS, which is well above the practical real-time threshold ($\geq$ 30 FPS). Thus, while the rendering-time overhead is non-negligible, our framework successfully balances enhanced view-dependent fidelity with interactive rendering speeds.

Analysis of View-Dependent Modeling. To further validate the effectiveness of our view-dependent refinement framework, we compare ViewSplat against the baseline SPFSplat with varying SH degrees (Tab. 3). While increasing the SH degree of the baseline model from 4 to 8 yields negligible improvement and may even slightly degrade performance, ViewSplat with SH degree 4 significantly outperforms all baseline variants. This suggests that standard SH-based representations reach a performance limit in pose-free settings, whereas our view-dependent refinement provides superior expressive capacity for capturing complex specularities and non-linear appearances.

Table 4: Cross-dataset generalization. All methods are trained on RE10K and evaluated in a zero-shot setting on ACID, DTU, and DL3DV. Results using the **ViewSplat** framework are highlighted in **bold**, demonstrating consistent performance gains across all SPFSplat variants.

Method	ACID			DTU			DL3DV		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Pose-Required									
pixelSplat	25.477	0.770	0.207	15.067	0.539	0.341	18.688	0.582	0.354
MVSplat	25.525	0.773	0.199	14.542	0.537	0.324	17.786	0.545	0.357
Supervised Pose-Free									
NoPoSplat*	25.764	0.776	0.199	17.899	0.629	0.279	19.974	0.612	0.305
Self-Supervised Pose-Free									
SelfSplat	22.204	0.686	0.316	13.249	0.434	0.441	15.047	0.410	0.498
PF3plat	20.726	0.610	0.308	12.972	0.407	0.464	15.773	0.458	0.417
SPFSplat	25.965	0.781	0.190	16.550	0.579	0.270	19.172	0.573	0.315
+ViewSplat	**26.595**	**0.793**	**0.179**	**16.864**	**0.589**	**0.259**	**19.858**	**0.591**	**0.298**
SPFSplatV2	26.220	0.789	0.185	16.793	0.584	0.265	19.439	0.584	0.304
+ViewSplat	**26.761**	**0.798**	**0.175**	**17.034**	**0.588**	**0.258**	**19.953**	**0.599**	**0.290**
SPFSplatV2-L	26.361	0.796	0.169	17.739	0.653	0.228	19.743	0.613	0.277
+ViewSplat	**27.169**	**0.810**	**0.154**	**18.174**	**0.665**	**0.215**	**20.563**	**0.635**	**0.256**

Cross-Dataset Generalization. To evaluate the generalization capability of our method, we conduct zero-shot evaluations on the ACID [20], DTU [12], and DL3DV [19] datasets after training exclusively on RE10K. DTU and DL3DV are



(a) Cross-Dataset Generalization: RE10K → ACID (b) Cross-Dataset Generalization: RE10K → DTU

Fig. 4: Qualitative results of cross-dataset generalization.

particularly challenging out-of-distribution settings for RE10K-trained models, due to their object-centric scenes and large-scale outdoor trajectories, respectively, which helps explain their lower absolute PSNR values. As shown in Tab. 4 and Fig. 4, ViewSplat consistently improves all SPFSplat variants across datasets, with the strongest performance obtained using the SPFSplatV2-L backbone. These results indicate that our view-dependent refinement learns transferable representations that generalize beyond the RE10K training distribution.

4.4 Ablation Studies

Effectiveness of Hypernetwork-style Formulation vs. Direct Regression. We compare our hypernetwork-style scene-conditioned View MLP method with a *Direct Regression* baseline that predicts pixel-wise Gaussian offsets from concatenated context features and a target pose (Tab. 5). Direct regression yields a small PSNR gain over ours (+0.08 dB), but reduces rendering speed from 154 to 72 FPS, corresponding to a **+436%** rendering-time overhead over SPFSplat. This is because the target pose is coupled with the heavy DPT decoder, requiring the dense network to be rerun for each target-view query.

In contrast, our hypernetwork formulation predicts per-pixel View MLP weights *once* during inference and applies only lightweight pose-conditioned View MLPs during rendering. Thus, ViewSplat preserves real-time rendering while achieving fidelity comparable to direct regression.

Table 5: Architectural design analysis on RE10K. We compare our hypernetwork-style scene-conditioned View MLP with a direct offset regression baseline.

Method	PSNR ↑	SSIM ↑	LPIPS ↓	Inference Time (s) ↓	Rendering Speed (FPS) ↑
SPFSplat	25.484	0.847	0.153	**0.046**	**386**
View MLP (Ours)	26.317	0.857	0.144	0.057	154
Direct Regression	**26.396**	**0.858**	**0.143**	0.048	72

Ablation on Refinement Components. Tab. 6 shows that most refinement subsets improve performance over the baseline. However, case (4) suffers from

catastrophic degradation when mean (μ) offsets are applied without corresponding opacity refinement. This decoupling leads to severe artifacts, blurring, and structural collapse, as shown in Fig. 5. These results confirm that spatial shifts must be synchronized with visibility updates to preserve structural coherence. By jointly optimizing all parameters, our full model (8) achieves the best performance across all metrics, underscoring the importance of holistic refinement.

Table 6: Effectiveness of refinement components. We evaluate the contribution of residual offsets for each Gaussian parameter ($\Delta\mu$, $\Delta\alpha$, Δr, Δs, and Δc) on RE10K. The **best** and second-best results are highlighted.

Method	$\Delta\mu$	$\Delta\alpha$	$\Delta(r,s)$	Δc	PSNR↑	SSIM↑	LPIPS↓
(1) Baseline (SPFSplat)	×	×	×	×	25.484	0.847	0.153
(2) $\Delta(\mu,\alpha)$ only	✓	✓	×	×	25.760	0.850	0.155
(3) $\Delta(r,s,c)$ only	×	×	✓	✓	26.241	0.856	0.146
(4) w/o $\Delta\alpha$	✓	×	✓	✓	15.684	0.555	0.525
(5) w/o $\Delta\mu$	×	✓	✓	✓	26.307	**0.857**	0.145
(6) w/o $\Delta(r,s)$	✓	✓	×	✓	26.282	**0.857**	0.145
(7) w/o Δc	✓	✓	✓	×	26.098	0.856	0.145
(8) Full (ViewSplat)	✓	✓	✓	✓	**26.317**	**0.857**	**0.144**

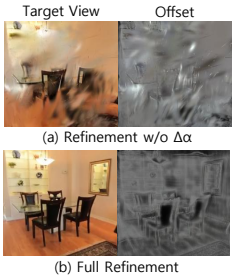


Fig. 5: Artifacts from decoupled μ/α refinement.

Effect of View MLP Hidden Dimension. As shown in Tab. 7, reducing the hidden dimension D of the View MLP improves rendering speed and memory efficiency with minimal quality loss. Decreasing D from 16 to 8 increases rendering speed from 154 to 186 FPS and reduces memory usage by 18.5%, while PSNR drops by only 0.003 dB. Further reducing D to 4 provides additional speedup but leads to a larger quality drop. Thus, while we use $D = 16$ in the main experiments for maximum fidelity, $D = 8$ offers the best trade-off between efficiency and quality. Additional analyses, including backbone generalization with YoNoSplat [36] and context-view scaling, are provided in the supplementary material.

Table 7: Quantitative evaluation of rendering quality and efficiency on RE10K with varying hidden dimensions (D) of the View MLP in our ViewSplat framework.

D	PSNR↑	SSIM↑	LPIPS↓	**Inf. Time (s)↓**	**FPS↑**	**Mem. (GiB)↓**
16	26.317	0.857	0.144	0.057	154	0.648
8	26.314	0.857	0.144	0.057	186	0.528
4	26.302	0.857	0.144	0.056	207	0.487

4.5 Limitations

Despite the improvements achieved by ViewSplat through its view-dependent refinement framework, it inherits a key limitation of the baseline SPFSplat: the inability to synthesize content in unobserved regions. Since both methods are

reconstruction-based and lack generative priors, they cannot hallucinate missing information, which results in blurry artifacts or empty regions in sparse-view areas, as illustrated in Fig. 6. This limitation highlights the inherent challenge of achieving complete scene recovery without incorporating generative modeling.

Furthermore, our view-adaptive design introduces an inherent trade-off between rendering quality and computational throughput. Because the Gaussian attributes are refined on the fly, conditioned on each specific target-view pose, the attributes must be re-evaluated for every novel viewpoint. This target-view pose conditioning prevents the framework from caching a single static 3D scene representation, leading to a substantial reduction in rendering frame rates compared to the static baseline (as quantified in Tab. 2). It also implies that multi-view consistency is maintained within a family of view-conditioned approximations rather than a single rigid geometry, which remains an important direction for future investigation.

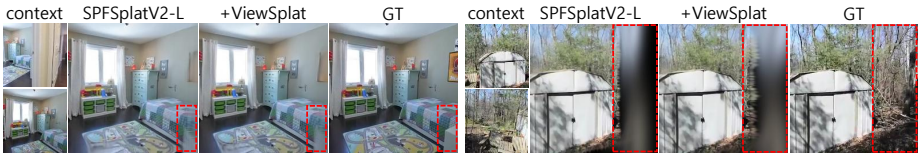


Fig. 6: Limitations: visual artifacts in unobserved regions.

5 Conclusion

We present ViewSplat, a view-adaptive framework that addresses the fidelity bottleneck in feed-forward 3D Gaussian splatting. By shifting from static primitive regression to view-adaptive splatting, our method refines Gaussian attributes conditioned on each target view, enabling the modeling of complex view-dependent appearance that rigid representations often fail to capture. ViewSplat achieves state-of-the-art fidelity while maintaining real-time rendering (90–154 FPS) across multiple benchmarks, without requiring precomputed poses. We hope our view-adaptive paradigm inspires future research into more flexible and high-fidelity 3D representations for real-world scenarios.

Acknowledgements

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (RS-2026-25481939) and the Industrial Core Technology Development Program of MOTIE/KEIT (RS-2025-02307680, Development of an inline vision inspection system based on the on-device AI semiconductor for real-time large-area defect detection). The authors acknowledge the Urban Big data and AI Institute of the University of Seoul supercomputing resources (http://ubai.uos.ac.kr) made available for conducting the research reported in this paper.

References

1. Boss, M., Braun, R., Jampani, V., Barron, J.T., Liu, C., Lensch, H.P.: Nerd: Neural reflectance decomposition from image collections. In: Int. Conf. Comput. Vis. pp. 12684–12694 (2021)
2. Boss, M., Jampani, V., Braun, R., Liu, C., Barron, J., Lensch, H.P.: Neural-pil: Neural pre-integrated lighting for reflectance decomposition. In: Adv. Neural Inform. Process. Syst. pp. 10691–10704 (2021)
3. Charatan, D., Li, S.L., Tagliasacchi, A., Sitzmann, V.: pixelsplat: 3d gaussian splats from image pairs for scalable generalizable 3d reconstruction. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 19457–19467 (2024)
4. Chen, S., Cavallari, T., Prisacariu, V.A., Brachmann, E.: Map-relative pose regression for visual re-localization. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 20665–20674 (2024)
5. Chen, Y., Xu, H., Zheng, C., Zhuang, B., Pollefeys, M., Geiger, A., Cham, T.J., Cai, J.: Mvsplat: Efficient 3d gaussian splatting from sparse multi-view images. In: Eur. Conf. Comput. Vis. pp. 370–386 (2024)
6. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: Transformers for image recognition at scale. In: Int. Conf. Learn. Represent. (2021)
7. Ha, D., Dai, A.M., Le, Q.V.: Hypernetworks. In: Int. Conf. Learn. Represent. (2017)
8. Hong, S., Jung, J., Shin, H., Han, J., Yang, J., Luo, C., Kim, S.: Pf3plat: Pose-free feed-forward 3d gaussian splatting. In: Int. Conf. Mach. Learn. (2025)
9. Hong, S., Jung, J., Shin, H., Yang, J., Kim, S., Luo, C.: Unifying correspondence pose and nerf for generalized pose-free novel view synthesis. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 20196–20206 (2024)
10. Huang, R., Mikolajczyk, K.: No pose at all: Self-supervised pose-free 3d gaussian splatting from sparse views. In: Int. Conf. Comput. Vis. pp. 27947–27957 (2025)
11. Huang, R., Mikolajczyk, K.: Spfsplatv2: Efficient self-supervised pose-free 3d gaussian splatting from sparse views (2025), arXiv preprint arXiv:2509.17246
12. Jensen, R., Dahl, A., Vogiatzis, G., Tola, E., Aanaes, H.: Large scale multi-view stereopsis evaluation. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 406–413 (2014)
13. Jiang, L., Mao, Y., Xu, L., Lu, T., Ren, K., Jin, Y., Xu, X., Yu, M., Pang, J., Zhao, F., et al.: Anysplat: Feed-forward 3d gaussian splatting from unconstrained views. ACM Trans. Graph. **44**(6), 1–16 (2025)
14. Jiang, Y., Tu, J., Liu, Y., Gao, X., Long, X., Wang, W., Ma, Y.: Gaussianshader: 3d gaussian splatting with shading functions for reflective surfaces. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 5322–5332 (2024)
15. Kang, G., Yoo, J., Park, J., Nam, S., Im, H., Shin, S., Kim, S., Park, E.: Selfsplat: Pose-free and 3d prior-free generalizable 3d gaussian splatting. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 22012–22022 (2025)
16. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. ACM Trans. Graph. **42**(4), 1–14 (2023)
17. Leroy, V., Cabon, Y., Revaud, J.: Grounding image matching in 3d with mast3r. In: Eur. Conf. Comput. Vis. pp. 71–91 (2024)
18. Liang, R., Chen, H., Li, C., Chen, F., Panneer, S., Vijaykumar, N.: Envidr: Implicit differentiable renderer with neural environment lighting. In: Int. Conf. Comput. Vis. pp. 79–89 (2023)

19. Ling, L., Sheng, Y., Tu, Z., Zhao, W., Xin, C., Wan, K., Yu, L., Guo, Q., Yu, Z., Lu, Y., Li, X., Sun, X., Ashok, R., Mukherjee, A., Kang, H., Kong, X., Hua, G., Zhang, T., Benes, B., Bera, A.: Dl3dv-10k: A large-scale scene dataset for deep learning-based 3d vision. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 22160–22169 (2024)
20. Liu, A., Tucker, R., Jampani, V., Makadia, A., Snavely, N., Kanazawa, A.: Infinite nature: Perpetual view generation of natural scenes from a single image. In: Int. Conf. Comput. Vis. pp. 14458–14467 (2021)
21. Liu, J., Tang, X., Cheng, F., Yang, R., Li, Z., Liu, J., Huang, Y., Lin, J., Liu, S., Wu, X., et al.: Mirrorgaussian: Reflecting 3d gaussians for reconstructing mirror reflections. In: Eur. Conf. Comput. Vis. pp. 377–393 (2024)
22. Liu, Y., Wang, P., Lin, C., Long, X., Wang, J., Liu, L., Komura, T., Wang, W.: Nero: Neural geometry and brdf reconstruction of reflective objects from multiview images. ACM Trans. Graph. **42**(4), 1–22 (2023)
23. Lu, T., Yu, M., Xu, L., Xiangli, Y., Wang, L., Lin, D., Dai, B.: Scaffold-gs: Structured 3d gaussians for view-adaptive rendering. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 20654–20664 (2024)
24. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. Communications of the ACM **65**(1), 99–106 (2021)
25. Ranftl, R., Bochkovskiy, A., Koltun, V.: Vision transformers for dense prediction. In: Int. Conf. Comput. Vis. pp. 12179–12188 (2021)
26. Schonberger, J.L., Frahm, J.M.: Structure-from-motion revisited. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 4104–4113 (2016)
27. Smart, B., Zheng, C., Laina, I., Prisacariu, V.A.: Splatt3r: Zero-shot gaussian splatting from uncalibrated image pairs (2024), arXiv preprint arXiv:2408.13912
28. Srinivasan, P.P., Deng, B., Zhang, X., Tancik, M., Mildenhall, B., Barron, J.T.: Nerv: Neural reflectance and visibility fields for relighting and view synthesis. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 7495–7504 (2021)
29. Tang, J., Chen, Z., Chen, X., Wang, T., Zeng, G., Liu, Z.: Lgm: Large multi-view gaussian model for high-resolution 3d content creation. In: Eur. Conf. Comput. Vis. pp. 1–18 (2024)
30. Verbin, D., Hedman, P., Mildenhall, B., Zickler, T., Barron, J.T., Srinivasan, P.P.: Ref-nerf: Structured view-dependent appearance for neural radiance fields. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 5481–5490 (2022)
31. Wang, J., Chen, M., Karaev, N., Vedaldi, A., Rupprecht, C., Novotny, D.: Vggt: Visual geometry grounded transformer. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 5294–5306 (2025)
32. Wang, S., Leroy, V., Cabon, Y., Chidlovskii, B., Revaud, J.: Dust3r: Geometric 3d vision made easy. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 20697–20709 (2024)
33. Wang, Z., Bovik, A., Sheikh, H., Simoncelli, E.: Image quality assessment: from error visibility to structural similarity. IEEE Trans. Image Process. **13**(4), 600–612 (2004)
34. Xu, Y., Shi, Z., Yifan, W., Chen, H., Yang, C., Peng, S., Shen, Y., Wetzstein, G.: Grm: Large gaussian reconstruction model for efficient 3d reconstruction and generation. In: Eur. Conf. Comput. Vis. pp. 1–20 (2024)
35. Yang, Z., Gao, X., Sun, Y.T., Huang, Y.H., Lyu, X., Zhou, W., Jiao, S., Qi, X., Jin, X.: Spec-gaussian: Anisotropic view-dependent appearance for 3d gaussian splatting. In: Adv. Neural Inform. Process. Syst. pp. 61192–61216 (2024)

36. Ye, B., Chen, B., Xu, H., Barath, D., Pollefeys, M.: Yonosplat: You only need one model for feedforward 3d gaussian splatting. In: Int. Conf. Learn. Represent. (2026)
37. Ye, B., Liu, S., Xu, H., Xueting, L., Pollefeys, M., Yang, M.H., Songyou, P.: No pose, no problem: Surprisingly simple 3d gaussian splats from sparse unposed images. In: Int. Conf. Learn. Represent. (2025)
38. Zhang, K., Bi, S., Tan, H., Xiangli, Y., Zhao, N., Sunkavalli, K., Xu, Z.: Gs-lrm: Large reconstruction model for 3d gaussian splatting. In: Eur. Conf. Comput. Vis. pp. 1–19 (2024)
39. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 586–595 (2018)
40. Zhang, X., Srinivasan, P.P., Deng, B., Debevec, P., Freeman, W.T., Barron, J.T.: Nerfactor: Neural factorization of shape and reflectance under an unknown illumination. ACM Trans. Graph. **40**(6), 1–18 (2021)
41. Zhou, T., Tucker, R., Flynn, J., Fyffe, G., Snavely, N.: Stereo magnification: Learning view synthesis using multiplane images. ACM Trans. Graph. **37**(4), 1–12 (2018)
42. Zhou, Y., Zhang, F., Wang, Z., Zhang, L.: Rtr-gs: 3d gaussian splatting for inverse rendering with radiance transfer and reflection. In: ACM Int. Conf. Multimedia. pp. 6888–6897 (2025)